REPORT DOCUMENTATION PAGE

Form Approved OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** April, 1997	**3. REPORT TYPE AND DATES COVERED** April 6-10, 1997

4. TITLE AND SUBTITLE
Parallel simulations of waste incineration in a dump combustor

5. FUNDING NUMBERS
supported by the ONR/SERDP under grant No. N00014-95-1-0163

6. AUTHOR(S)
S. Arunajatesan, S. Menon

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0150

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
SERDP
901 North Stuart St. Suite 303
Arlington, VA 22203

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
N/A

11. SUPPLEMENTARY NOTES
The United States Government has a royalty-free license throughout the world in all copyrightable material contained herein. All other rights are reserved by the copyright owner.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release: distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT *(Maximum 200 Words)*
A parallel code has been developed and used to study the waste incineration in a dump combustor. Results from some timing tests to characterize the performance of the code on massively parallel machines like the Intel-Paragon, IBM-SP2 and the Cray-T3D and workstation class machines like the SGI Power Challenge are presented. It is shown that the performance of the code can be significantly enhanced by increasing the computation to communication work load ratio per processor. Some results from the waste incineration studies are presented which demonstrate the possibility of increasing waste consumption in the dump combustor by acoustically and mechanically forcing the inlet and fuel streams.

14. SUBJECT TERMS
SERDP; SERDP Collection;

15. NUMBER OF PAGES

16. PRICE CODE N/A

17. SECURITY CLASSIFICATION OF REPORT unclass.	**18. SECURITY CLASSIFICATION OF THIS PAGE** unclass.	**19. SECURITY CLASSIFICATION OF ABSTRACT** unclass.	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

DTIC QUALITY INSPECTED 4

19990521 138

cf·[illegible]
cf· 887

PARALLEL SIMULATIONS OF WASTE INCINERATION IN A DUMP COMBUSTOR

S. Arunajatesan and S. Menon
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta,GA 30332-0150

Proceedings of
High Performance Computing 1997,
Grand Challenges in Computer Simulation,
Atlanta, GA, April 6-10, 1997, pp 151-156.

PARALLEL SIMULATIONS OF WASTE INCINERATION IN A DUMP COMBUSTOR

S. Arunajatesan and S. Menon
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta,GA 30332-0150

Abstract

A parallel code has been developed and used to study the waste incineration in a dump combustor. Results from some timing tests to characterize the performance of the code on massively parallel machines like the Intel-Paragon, IBM-SP2 and the Cray-T3D and workstation class machines like the SGI Power Challenge are presented. It is shown that the performance of the code can be significantly enhanced by increasing the computation to communication work load ratio per processor. Some results from the waste incineration studies are presented which demonstrate the possibility of increasing waste consumption in the dump combustor by acoustically and mechanically forcing the inlet and fuel streams.

1 Introduction

Simulation of realistic turbulent reacting fluid flow problems demands large amounts of computer resources, since, to resolve the vast range of scales of motion in high Reynolds turbulent flows requires very high grid resolutions. This, coupled with the need to simulate the chemical reactions accurately to capture the turbulence-chemistry interactions in these flows, makes the simulation of these problems on conventional single processor systems almost impossible for realistic flow parameters.

However, with the advent of Massively Parallel computers, it has become possible to simulate these large scale problems with reasonable turnaround time. The vast computational domain is split up among the large number of processors, thus enabling each processor to work on a smaller domain, and hence, perform better. However, this also requires communication between the processors, which is accomplished by explicit and implicit message passing algorithms. Message passing represents an extra overhead for any simulation, since, a large amount of inter-processor communication can actually slow down the simulation to a level that it is almost unrealistically expensive. Thus, using the massively parallel systems for these large fluid flow problems involves balancing the speed-up obtained by decomposing the problem into smaller sub-domains with the additional overhead due to the message passing.

In this paper, we present some results from a study of combustion processes in a dump combustor which is part of a larger program to develop efficient compact waste incinerators. Gaseous fuel and waste surrogates are injected into the shear layer formed behind the sudden expansion in a dump combustor. A 12-species-12-reaction reduced finite rate kinetics model is employed for the ethylene-benzene chemistry. The mechanism has been validated against a 277 reaction full kinetics model,[1]. The efficiency of the waste destruction is studied with a view to improve the performance of the combustor and to develop control algorithms to achieve better destruction rates and greater scalability. Another goal of this research effort is to develop computationally cheaper models for simulation of these flows in a design environment. The data obtained from the detailed simulations will be used to help formulate and calibrate the model.

2 Computational method

In order to simulate the flow in a dump combustor, the governing equations of axisymmetric multi-component reacting flows are solved numerically on a discrete grid. This is done using a fourth order extension of the MacCormack scheme proposed by Bayliss et al.[2]. The unsteady equations are marched in time using the second order Runge-Kutta scheme. At the boundaries, the accuracy of the scheme is reduced to second order. The boundary conditions at the inflow and outflow are handled using characteristic boundary conditions. No slip and adiabatic conditions are imposed at the walls and symmetry conditions are imposed on the centerline. A schematic of the geometry used in these simulations is shown in figure 1.

For the reacting flow cases, the chemical source term in the governing equations are solved using an implicit method. This is done to overcome the stiffness problem due to the vastly different time scales of the chemistry and the fluid mechanics. In some cases, an explicit method is also shown for comparison (in terms of the extra expense involved).

3 Parallel implementation

The code described above was implemented on multiprocessor parallel machines using the domain decomposition technique (figure 2). In this method, the computational domain is broken down into two domains. Each domain is then broken down further into smaller sub-domains. Each one of these smaller sub-domains is then solved on one processor. However, the fourth order numerical scheme requires information from adjacent points for the solution at a given point. So, a small overlap of computational cells is maintained between the processors. At each time step, the information on these overlap cells is updated through explicit message passing. This is done using the message passing libraries of MPI [3]. This library contains various routines that can be used to exchange information between the processors. It also has the advantage that it is completely portable to different machines and enables a single code to be used on a variety of machines with little or no changes to the code. This code has been successfully implemented on the Intel-Paragon, the IBM-SP2, the Cray-T3D and an SGI Power Challenge array.

Since the domain sizes are different in the inlet and combustor regions, different processor distributions are used in the two regions. However the overlap between the processors needs to be maintained across this boundary for computational accuracy. This is accomplished by specialized message passing between the two domains,[4]. The details of the cell partitioning and the communication methodology are explained in detail in that paper and the reader is referred to it for further details. In this paper, we attempt to characterize the optimal performance range of the machines under consideration for the chosen problem.

4 Performance Characteristics

In this section we discuss some of the timing measurements and the implications of the results therein. The results shown here are from timing tests conducted primarily on the Intel-Paragon and the Cray-T3D. Some results from tests on the IBM-SP2 are also shown.

4.1 Load Balance

In order to do reliable timing studies using such a code, it is important to ensure that all the processors are doing approximately the same amount of work. In a practical simulation of the kind discussed here, it is almost impossible to make sure that exactly the same amount of work is performed by each node due to the presence of boundaries and corner points for which special calculations need to be performed. However, relative load imbalance, as defined in equation 1, is a good measure of any imbalance in the work load of the nodes.

$$RelativeLoadImbalance[RLI] = \left|\frac{T_{comp,i} - T_{comp,av.}}{T_{comp,av.}}\right| \tag{1}$$

where, $T_{comp,i}$ is the computational time for the i^{th} processor and $T_{comp,av.}$ is the average computational time over all the processors.

A plot of the RLI is shown in figure 3. It can be seen that the total work load is fairly evenly distributed among all the nodes. The noticeable peaks at processor numbers 12, 16, 20, 24 are due to the boundary conditions. The boundary conditions in this code are calculated using the characteristic formulation of Poinsot and Lele [5]. This involves the separate solution of the complete system of equations at the outflow boundary nodes. This can be very expensive when the ratio of the number of boundary to internal nodes is large. This is seen from figure 3. As the number of internal nodes is increased, this ratio drops and the relative expense to do these boundary computations drops. Hence, a better load balance is achieved at larger grid nodes per processor ratios. This fact is also seen from figure 4. In this plot the RLI is shown for the 8^2 and 12^2 grids with and without chemistry. In the presence of chemistry, the computational expense for the internal nodes increases greatly. This causes the expense of the boundary computations to drop relative to that of the internal computations. Hence, again, a better load balance is achieved.

4.2 Parallel Efficiency

In this section we discuss the measurements of the efficiency of parallelisation of the code. In order to do this we need to define a measure of the efficiency. If the computational domain were rectangular, a simple measure would be

$$\eta = \frac{T_1}{NT_N} \tag{2}$$

where T_1 is the computation time using one processor and similarly, T_N is the computation time using N processors. However, in the present case, atleast two processors are required to perform the baseline computation. Also, for all the computations, load balance has to be maintained. This presents a fairly complex problem when taken with the restrictions on the processor assignments possible on the machines under consideration. Hence we just present the raw timing data in terms of CPU seconds per timestep.

In order to appreciate the effects of the two main conflicting parameters in such a simulation (as discussed earlier), we show the results of a preliminary scale up study conducted on the Intel-Paragon, IBM-SP2 and the Cray-T3D in figures 5, 6, and 7. These tests were conducted to study the scale up achieved with the increase in number of processors and problem size. The tests were conducted on a grid size of 288 by 96 points. Three different cases were tested: flow without any chemistry; flow with pure scalar mixing, and flow with full finite rate chemistry. The two latter cases involve six additional scalar equations.

The best measure of the efficiency of a parallel simulation is the scale up achieved by increasing the number of processors. Ideally, doubling the number of processors should result in a doubling of the speed of the code. As was discussed earlier, there are two conflicting factors affecting this in simulations of the kind described here. The gain due to the reduction in problem size per processor is in part canceled by the extra overhead due to additional message passing. This is clearly visible in the data for the IBM-SP2. It can be seen that the CPU time per time step increases after a point with increase in number of processors. The optimum number of processors lies around 32 for this configuration. This suggests that the IBM-SP2 might be a good choice for a computationally intensive problem as opposed to a communication intensive problem.

This conflict is not very clearly visible in the case of the data for Intel-Paragon and the Cray-T3D. However, it can be seen that the phenomenon of diminishing returns has started to set in. Due to the limitation in the size of the machine, the tests could not be conducted on larger configurations. In none of the machines, the ideal linear scaling is realized. However, the Intel-Paragon comes very close to it.

From the above plots it is clear that as the communication overheads are lowered compared to the computation work, the performance of the code should improve. In order to verify this, we conducted further tests in which the processor grid assignment was varied from 8^2 grid per processor to 64^2 grid per processor. The results of these tests for the Intel-Paragon, Cray-T3D and the SGI Power Challenge cluster are shown in figure 8. These plots show the mean ratios over all the processors for each configuration. In all the cases, load balance is maintained so that the deviations in the timings from the averages are small. It is clearly seen that as the number of nodes per processor is increased, the ratio of the computation to communication time decreases. The Intel-Paragon seems to have the best ratios of the three machines. However, the SGI machines show rapid growth in the ratio with increase in the computation to communication work ratio.

Another measure of this increased performance with increase in the computation to communication work ratio is the processor utilization factor ϕ. It is defined as

$$\phi = \frac{T_{calc}}{T_{calc} + T_{comm}} \tag{3}$$

A plot of ϕ for the various processor grids is shown in figure 9. It can be seen that all the machines show the expected asymptotic behavior. However, an interesting feature is the rapid increase in ϕ with increase in cells per processor for the SGI-PC machine. This is indicative of the fact that the SGI-PC may be more suitable for very largely computing intensive simulations when compared to the other machines. This fact is further borne out by the next figure. Here, (figure 10), we show the computing time per cell per timestep for each of the three machines. It is clear that the SGI-PC is almost an order of magnitude faster than the other machines. However, in order to translate this speed into performance, it is necessary to increase the processor utilization. The authors feel that comparisons using higher number of cells per processor would indicate this more clearly, however, limitations of available memory prevent such evaluations at the present time.

Another important feature of turbulent reacting flow simulations is the increase in cost when the number of scalars or species in the simulations is increased. This is due to the fact that a larger system of equations needs to be solved. When chemical reactions are involved the cost escalates further due to the increased cost of accurately solving the stiff equations typically associated with these problems. This feature is also highlighted in figure 10. It is seen that the cost per cell goes up by more than an order of magnitude for the case with chemical reactions. This would suggest that for such problems, better performance may be obtained by using a machine with more computing power, like perhaps, the SGI-PC.

5 Results

We now present some of the simulation results obtained by using the code described above. The results presented here are those obtained from a study of the waste incineration process in a dump combustor. A more detailed discussion of the results and the associated physical processes is presented in [1] and the reader is referred to that paper for details.

A gaseous waste surrogate and fuel mixture is injected into the shear layer behind the dump in order to increase the residence time of the injected species and enhance the consumption efficiency. It was found that acoustically and mechanically forcing the flow resulted in enhanced consumption rates in the dump combustor.

A plot of the Waste (benzene) Destruction and Removal Efficiency (DRE) versus the axial distance is shown in figure 11. The DRE is measured in terms of number of nines which is indicative of the extent of waste consumption at any location [1]. Also shown in the same figure is the effect of forcing the inlet and fuel streams. The frequency of the forcing is set equal to the frequency of the vortex shedding at the dump plane. It is seen that a very significant increase in the consumption efficiency is obtained by forcing the inlet and fuel streams. Forcing the flow increases the intensity of the turbulence in the near field and this results in enhanced mixing rates in the near field of the dump plane. This enhanced mixing results in more efficient consumption of the injected waste. This enhanced mixing is due to the presence of a large number of small counter rotating vortices which aid in the transport of the fuel and air into the mixing zones. This greater mixing causes an advancement in the combustion zone as observed in the OH concentration fields, shown in figure 12.

Here, the radially integrated OH concentration at each axial location across the flame width is shown. The data is normalized by the OH concentration in the unforced case at $X/D = 3.0$. It is seen clearly that, in the unforced case, the OH concentration levels are very low in the region $X/D < 2.0$ and it suddenly shoots up in the periphery of the first roll-up. In contrast to this, for the forced cases, the OH concentration increases steadily right from the dump plane. The large OH production in the initial region indicates very rapid consumption of the fuel and waste surrogate in this region. This high consumption, in fact, causes a slight decrease in the OH concentration in the core of the first roll-up. Similar results were also observed in forced free shear layers by Gutmark et al. [6]. It should be noted that the combustion zone for the third case, where the fuel and waste are also forced is even closer to the dump plane than the other two cases. This fact is also reflected in the integrated OH concentration profiles.

6 Conclusion

A parallel code was developed and successfully used to study the process of waste incineration in a dump combustor. The code showed very good performance on large parallel machines like the Intel-Paragon, Cray-T3D and the SGI-PC class of machines. The code was used to study the scale up properties on these machines. It was found that significant gains in performance can be obtained by increasing the computation to communication work ratio per processor. It was also found that the SGI-PC and the IBM-SP2 show poor performance at lower cells per processor values, but this changes appreciably at larger number of cells per processor. It was also found that for communication intensive computations, the Intel-Paragon out-performs all the other machines tested.

Acknowledgments

This work was supported by the Office of Naval Research/SERDP Program under grant No. N00014-95-1-0163. The simulations were performed on the DOD high performance computer systems at Wright Patterson AFB (Intel-Paragon), Eglin AFB (Cray-T3D) and the Maui high performance computing center (IBM-SP2). These contributions are gratefully acknowledged.

References

[1] . Arunajatesan, S., and Menon, S., "Simulation of Controlled Injection of Toxic Waste for Enhanced Destruction in a Compact Incinerator", AIAA-97-3075, 1996.

[2] . Bayliss, A.,Parikh, P.,Maestrello, L.,Turkel, E., "A Fourth-Order Scheme for the Unsteady Compressible Navier-Stokes Equations", NASA-CR-177994, 1985.

[3] . "MPI: A Message-Passing Interface Standard", Message Passing Interface Forum, University of Tennessee, Knoxville, 1994.

[4] Weeratunga, S., Menon, S., "Parallel Computations of Unsteady Combustion in a Ramjet Engine", AIAA-93-1914, 1993.

[5] Poinsot, T.J., and Lele, S.K., "Boundary Conditions for Direct Simulations of Compressible Viscous Flows", Journal of Computational Physics, Vol. 101, pp 104-129, 1992.

[6] Gutmark, E., Parr, T.P., Hanson-Parr, D.M., Schadow, K.C., "On the Role of Large and Small-Scale Structures in Combustion Control", Combustion Science and Technology, Vol. 66, pp 107-126, 1982.

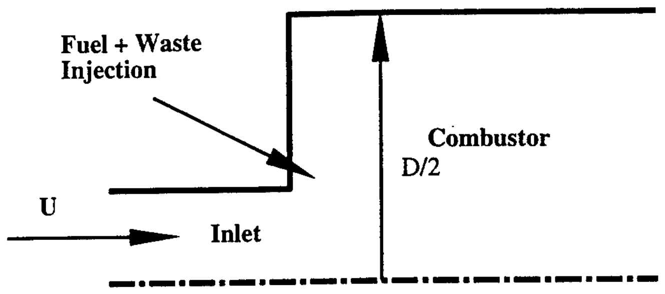

Fig. 1. Schematic of the Dump Combustor Geometry used in the present simulations.

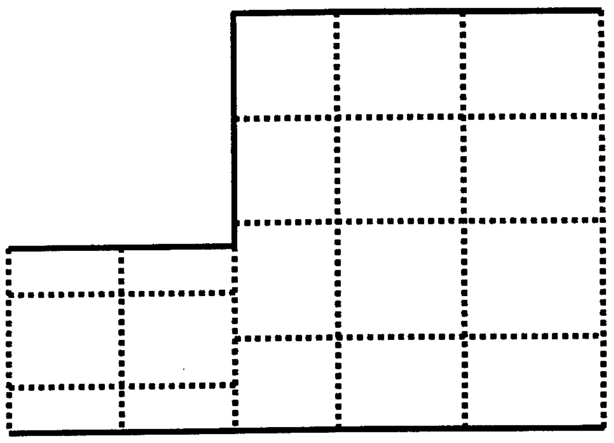

Fig. 2. Schematic of the Domain Decomposition used in the present parallelisation scheme.

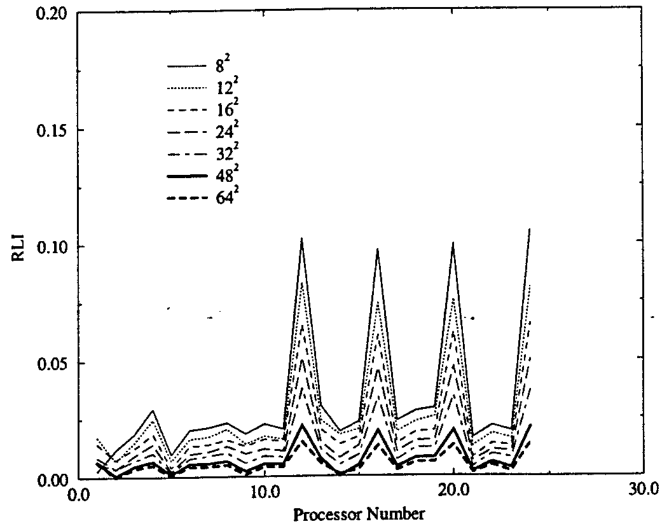

Fig. 3. A plot of the Relative Load Imbalance for various processor compute loads.

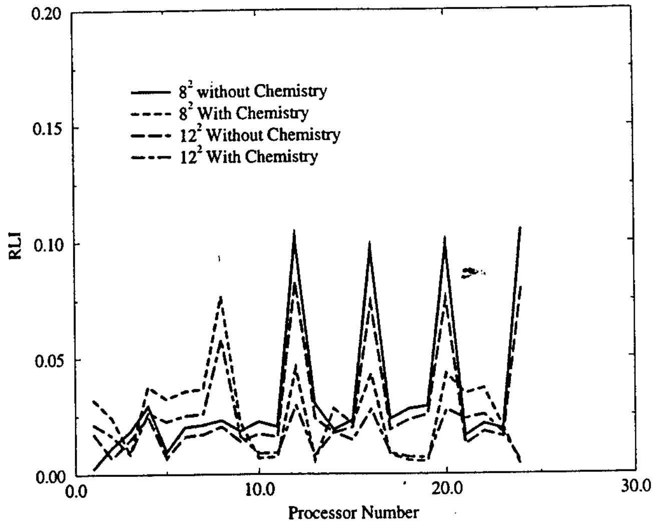

Fig. 4. A Comparison of the Relative Load Imbalance with and without chemistry.

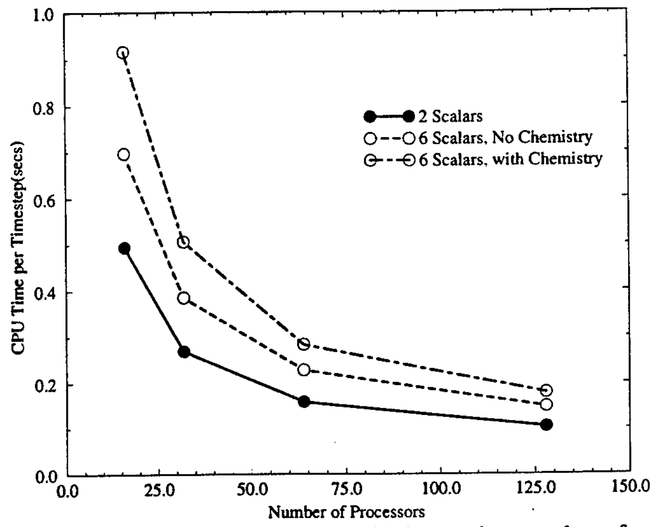

Fig. 5. CPU time per time step with increasing number of processors on the Intel–Paragon.

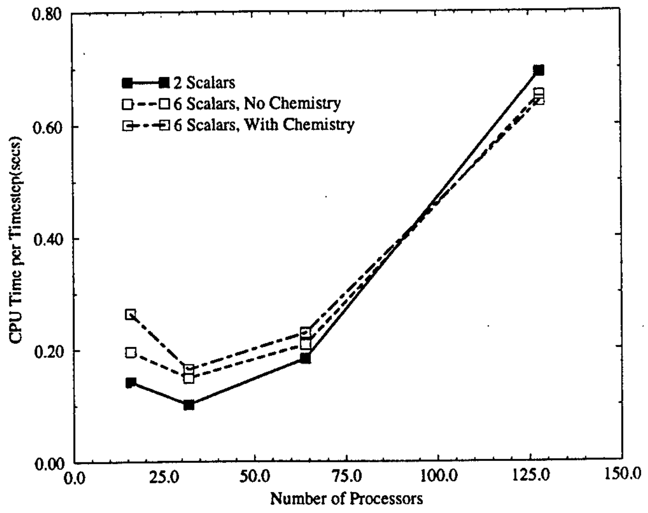

Fig. 6. CPU time per time step with increasing number of processors on the IBM–SP2.

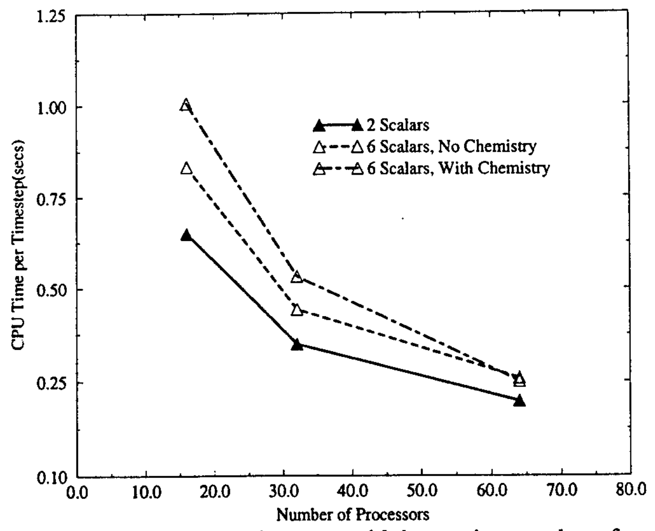

Fig. 7. CPU time per time step with increasing number of processors on theCray–T3D.

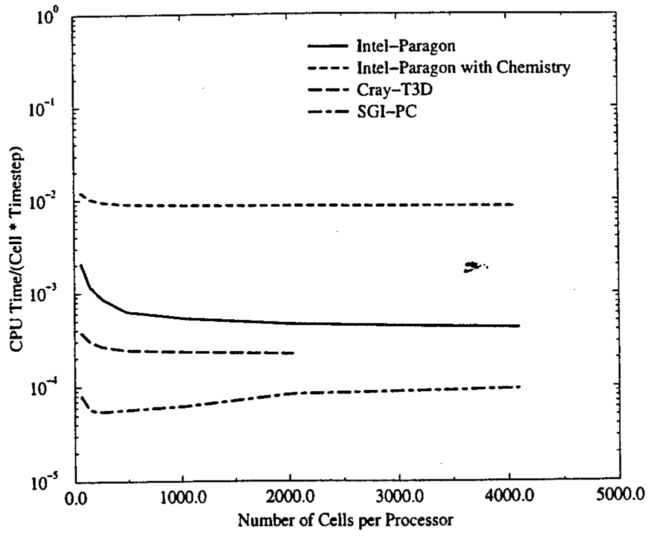

Fig. 10. Comparison of CPU times per cell per timestep for the various machines used in the present work.

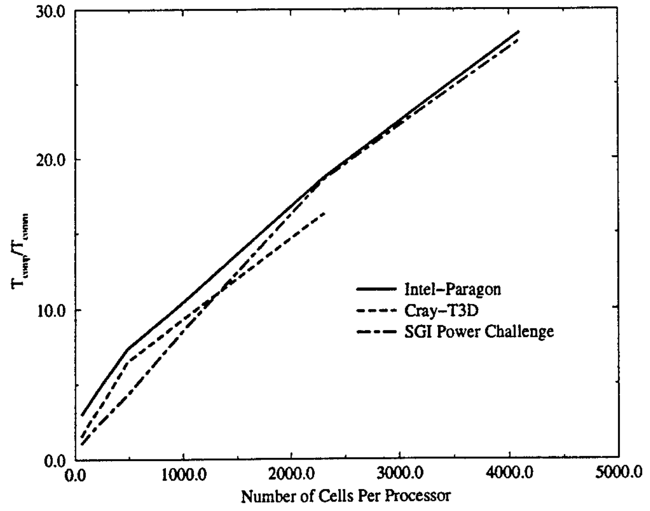

Fig. 8.Variation in the Computation to Communication time ratio with increasing number of computational cells per processor.

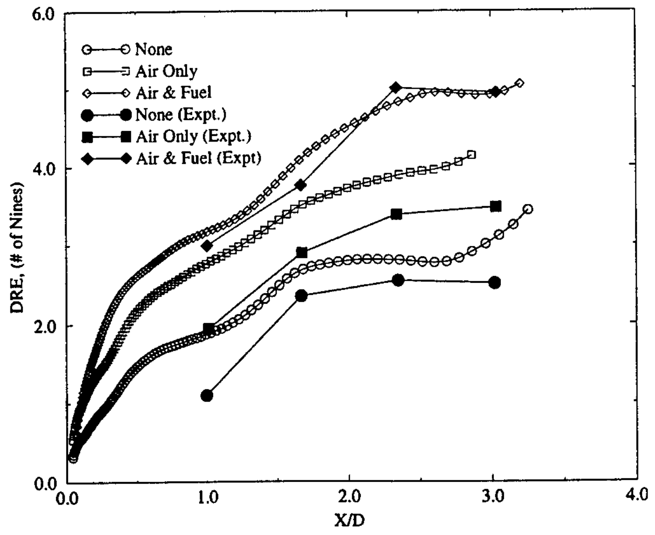

Fig. 11.Benzene DREs as a function of Axial Distance.

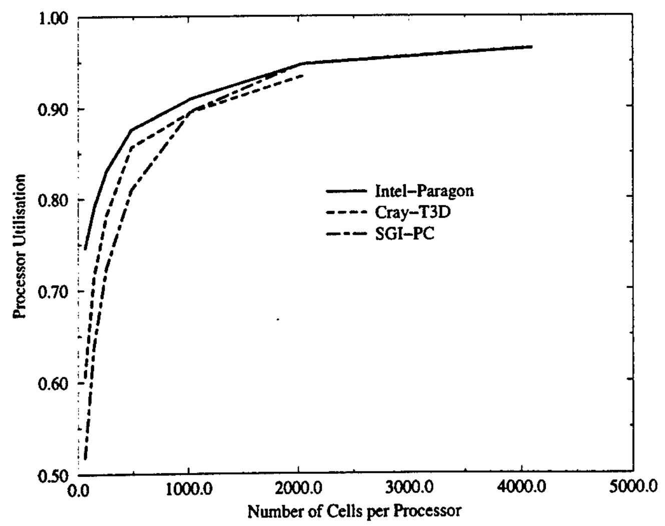

Fig. 9.Variation in the Processor Utilization factor with increasing number of computational cells per processor.

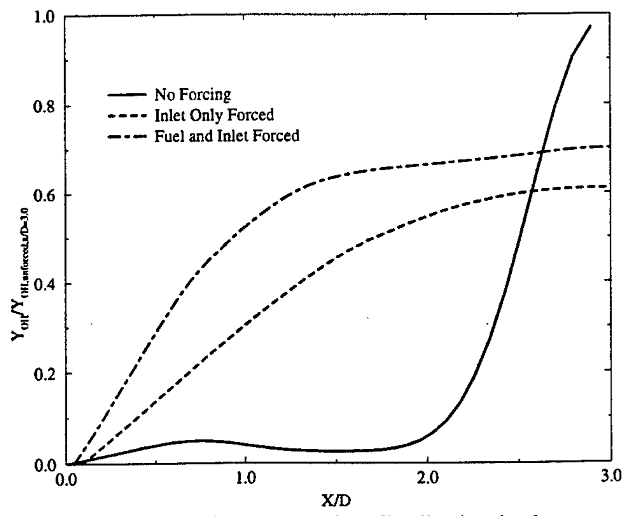

Fig. 12.Integated OH concentration distribution in the near field of the dump.